Implementing Secure Network Management Comprehending SNMPv3 Security

By Shawn Routhier, Wind River Systems and Chris Wellens, InterWorking Labs

Purpose

Our goal is to help computer users, network administrators, and security specialists understand the components of SNMPv3 security and how they work together. This paper does not discuss the pros and cons of any of the security algorithms commonly used with SNMPv3 or their underlying mathematics. Information on the security algorithms - DES, AES, MD5, SHA, etc. - may be found in many other documents on the web.

What’s Inside...

- Purpose (pg. 1)
- Introduction (pg. 1)
- What was wrong with security in SNMP? (pg. 2)
- How does SNMPv3 fix the security problems? (pg. 2)
- Configuring the engine ID (pg. 3)
- Configuring users for the SNMPv3 device (pg. 3)
- How does a user access the SNMPv3 Agent? (pg. 4)
- The first part: authentication (pg. 4)
- The second part: privacy/encryption (pg. 5)
- Reasons to Use or Not Use Privacy (pg. 6)
- Special Case Devices Use ContextEngineID and ContextName (pg. 6)
- How does the SNMPv3 agent process the message from the management application? (pg. 7)
- Conclusion (pg. 8)

Introduction

Many users of the Simple Network Management Program (SNMP) have been confused by the new security mechanisms in SNMPv3. They know that SNMPv3 is more secure than the previous versions, SNMPv1 and SNMPv2c, but may not know how to use the security features of SNMPv3. Below we summarize the essential security changes and help you understand the new vocabulary and new concepts in SNMPv3 security.
What was wrong with security in SNMP?

SNMPv1 and v2c used "community strings". There was a "read" community string and a "write" community string. (There could also be multiple read/write community strings, but typically this is not implemented.) If you knew the value of the read community string, you could enter that value when prompted, and read the values of MIB objects in the SNMP agent. If you knew and entered the value for the write community string you could enter that value when prompted and make changes to the values of writable MIB objects. The system administrator would give a user one or both community string values depending on the person's "need-to-know" and/or trustworthiness. Having the "write" community string implied a higher level of trust.

Most manufacturers used the defaults "public" for the read community string and "private" for the write community string. Many operators and network managers installed products without changing these community strings. Thus, an outsider could guess the values and potentially damage or disrupt a network.

Additionally, the values for the community strings were sent "in the clear" on the wire, so anyone with access to the network and a Sniffer or similar product could read the traffic passing over the network and determine the value of the community strings, and thus gain access to the network.

In summary, most of the security problems with SNMPv1 and v2c were due to the sending of community strings in the clear.

How does SNMPv3 fix the security problems?

SNMPv3 addresses security problems by adding two new features:

(1) Authentication via hashing and time stamps.

(2) Confidentiality via encryption.

SNMPv3 also introduced a new concept of the command generator and the command responder. These terms replace the traditional notion of a smart manager and a non-intelligent agent. SNMPv3 recognizes that many network devices and applications operate in dual or multiple modes. These devices are now "entities" and an entity can be a command generator (used to be a manager) or a command responder (used to be an agent).

SNMPv3 offers three new models for securing the access to management information. These models are listed below:

No Authentication and No Encryption. Using this option provides no security, confidentiality, or privacy at all. It might be useful for certain applications such as development and debugging to turn security off. This is called "no auth/no priv".
Authentication and No Encryption. The user at the management application is authenticated by the SNMPv3 entity before the entity allows the user to access any of the values in the MIB objects on the agent. This is call "auth/no priv".

Authentication and Encryption. The user at the management application is authenticated by the SNMPv3 entity before the entity allows the user to access any of the values in the MIB objects on the agent. In addition, all of the requests and responses from the management application to the SNMPv3 entity are encrypted, so that all the data is completely secure. This is called "auth/priv".

It is important to note that authentication and privacy are bi-directional. In other words, the SNMPv3 entity asking for the information is authenticated by the SNMPv3 entity providing the information. When the SNMPv3 entity receives the information, it authenticates the provider of the information. An alternative way of thinking about this is that no SNMPv3 entity trusts another SNMPv3 entity.

Now, let’s take a look at what a network administrator has to do to set up SNMPv3.

**Configuring the engine ID**

Each SNMPv3 agent has an engine ID that uniquely identifies the agent in the device. The engine ID may be set by the network administrator and is unique to that internal network, alternatively, it may be pre-configured by the device manufacturer.

Each packet contains two engine IDs. One is used to identify security information - user name, key location. The second one specifies where the packet payload is coming from and going to. (There will be more discussion on this later.)

SNMPv1 and v2c did not have such a concept of an engine ID; they relied instead on the IP address or the domain name of the device. However, IP addresses may be changed and the Domain Name System may be down. In addition, IP addresses and domain names may be known outside the organization. All of this can lead to security vulnerabilities.

Some devices may use proxies or contain multiple SNMP agents. A single device could have multiple engine IDs, each one associated with one of the SNMP agents in the device.

**Configuring Users for the SNMPv3 Device**

For each SNMPv3 device on the organization's network, the network administrator must create a "user login account" for the device. This is very similar to the process of setting up users to log into a Windows or Unix machine. The network administrator also has to decide what privileges each user has - either individually or as part of a group.
Here is an example. For the backbone routers, the network administrator may decide that only Rajiv and Xiang should be able to see all the MIB objects and make changes to the writable MIB objects. Thus, the network administrator configures those users for a full view of the MIB objects and full privileges. Alternatively, the network administrator could define a group called "wheel", configure the group in this manner, and then simply add Rajiv and Xiang to the group.

Next, the network administrator may decide that Dinesh, Sriniv, and Juergen should be able to monitor all activity for this router; he configures these users to view all the MIB objects in MIB-II, including all the interface tables, but they are not allowed to make any changes to writable objects. Alternatively the network administrator could define a group called "operator", configure the group in this manner, and then simply add Dinesh, Sriniv, and Juergen to the group.

Fortunately tools and wizards are available that make it easy to configure users and groups for an SNMPv3 device.

Now the users are set up …

**How does a user access the SNMPv3 agent?**

In most cases, the users will access the SNMPv3 agent by logging in to a management application on a host system. After entering their login information, users will expect to manage the MIB objects in the agent. Now we get back to that problem about sending passwords "in the clear" on the network. How are we going to handle that?

**The first part: authentication**

There are three goals with authentication: (1) to verify that the user is really who he says he is, (2) to verify the user’s message was not changed during transport, and (3) to verify that the message is not being replayed (copy the message and play it over again in the future).

Authentication is optional in SNMPv3. There may be situations where the users are trusted and/or the data is not sensitive. In these cases, the organization may not want to incur the overhead of using authentication. This is referred to as noauth/nopriv.

The user will enter his user name and authentication password in the management application. However, rather than having that information be sent "in the clear" on the network, something else will happen.

The management application will convert the authentication password into a "localized" key.

The key is formed by using a hashing function and two values. The value of the snmpEngineID of the SNMPv3 device and the user's password are combined via the hashing function and transformed into the "localized key".
At this point, the localized key, the time stamp from the SNMP engine on the agent and the actual message are transformed by some special security processing. The transformation results in a "message digest" (the "fingerprint" of the original message). Then the "message digest" is inserted into the original message and the combination is sent over the wire. Diagram A illustrates this message authentication process.

Diagram A: An Authenticated SNMPv3 Message

When the agent receives the message, the agent will authenticate the sender by checking if the message digest contained in the message is the same value that it computes itself using the same authentication key as the sender (See RFC 1321 for a description of one digest.).

In this way, unauthorized users cannot gain access to the agent since only the authorized user and the agent know the authentication key. In other words, if you do not know the authentication password, then you would not be able to produce the correct message digest expected by the agent and thus gain access to the agent. The most important thing is that the password itself is never sent "in the clear" over the wire as in the case of v1 and v2c.

The second part: privacy/encryption

The goal with privacy is to support confidentiality of the data that is transmitted over the wire. In order to make the data private and confidential, it needs to be encrypted.

Privacy is optional in SNMPv3. It is possible to use SNMPv3 in the mode called "auth, nopriv" which means using authentication, but not privacy. In many cases, the organization is not concerned with protecting information about the status of its network, such as a printer sending an "out of paper" condition.

If the privacy component of the SNMPv3 architecture is used, then the network administrator will provide each user a privacy password. When the user actually wants to access an SNMPv3 device, he will enter the user name assigned to him, the authentication password, and the privacy password into the management application.

Just as in authentication, the management application will convert the privacy password into a "localized key". The localized privacy key is used to encrypt the Scoped PDU portion in an SNMPv3 message --- The Protocol Data Unit (PDU, which contains the management commands) together with the contextEngineID and contextName in an SNMPv3 message constitute a "scopedPDU". Since the "scoped PDU" is encrypted prior to being transmitted, the
contents of the scopedPDU are now private. Diagram B illustrates this encryption process.

![Diagram B: An Encrypted SNMPv3 Message](image)

If you are using privacy, you must have authentication. There is no option of having privacy without authentication. It does not make sense to receive encrypted, confidential data, and then decrypt it, if you are not sure of the source of the information. Thus, authentication is considered a pre-requisite for privacy.

**Reasons to Use or Not Use Privacy**

Why would you not use privacy if it were available to you? Why make these options? Surely everyone wants his or her data private, right? Adding privacy to SNMPv3 messages adds overhead to the processing and the number of packets transmitted. There is always a cost in terms of using more network resources. It may be that adding privacy is not worth the cost.

**Special Case Devices Use ContextEngineID and ContextName**

SNMPv3 introduced a new architecture to allow for complex devices. Normally we think of an SNMP agent performing one set of functions for one device. Consider the case of one physical device with multiple "logical devices" embedded inside it.

![Diagram C: Physical router with one logical router, one logical switch, and one physical UPS](image)

In Diagram C, we have the case of a high-end router. This router has its own IP address. The router has a built-in UPS, and it has two logical switches A and B.

It is possible for logical switch A to fail while the logical switch B continues to run. In a situation like this, the MIB objects that refer to the performance of the device have to distinguish between logical switch A and logical switch B. Thus, it may be appropriate for each logical device to have its own set of MIB objects relating to its performance. If
logical switch A crashes and reboots, for example, but logical switch B continues, sysUpTime for the logical switch A would have the value of zero after a reboot, but the value of sysUpTime for the logical switch B would be a much larger number.

To address situations like this, SNMPv3 offers the contextName and contextEngineID. The physical router would have one contextName for the logical switch A and another contextName for logical switch B. A user or a management application could then specify the context name to get the performance information from the MIB objects associated with that contextName. The contextEngineID is the same as the engine ID of the target entity - in this example, the router.

The contextEngineID identifies the SNMP entity that should process the request (the physical router) and the contextName tells the agent in which context (logical switch A or B) it should search for the objects requested by the user or the management application.

Recall that there are two EngineIDs in a packet. The first one specifies where the security information association is processed. The second specifies where the payload is processed. The second one is the contextEngineID. Within a given context, the contextName specifies the logical entity that is used.

In our example, the company that built the router built a UPS inside the router enclosure. This UPS was acquired from another company and the UPS only supports SNMPv1. In this case, the UPS would have its own unique contextEngineID, but no external IP address. If the user or management application wanted some management information from the UPS, how does it find the UPS? The request would use the EngineID of the router and the contextEngineID of the UPS device. Now our SNMPv1 UPS that is in reality legacy equipment, enjoys the benefit of SNMPv3 security because it is responding to requests that are passed through by the router. The router, of course, has SNMPv3 fully implemented.

Note that some experts believe that the addition of contexts is a dangerous short cut for poor MIB design. The thinking is that an extremely well designed MIB would handle all the statistics in the agent by unique Object IDs, indices, and instances.

**How does the SNMPv3 agent process the message from the management application?**

When the SNMPv3 entity (agent) receives the message, the agent will authenticate the sender by checking if the message digest contained in the message is the same value that it computes itself using the same authentication key as the sender.

Once the SNMPv3 entity has authenticated the message, the next step is to decrypt the message if the message is encrypted.
If the SNMPv3 entity (agent) has the correct localized privacy key for the given user, then special security processing can go forward to decrypt the packet properly. If it does not have the correct privacy key then the packet would not be decrypted properly. In this case, privacy is assured because the receiving SNMPv3 entity (agent) does not have proper credentials.

Diagram D illustrates the authentication and privacy processing flow.

**Conclusion**

As you can see, SNMPv3 is not that hard. In fact, it is straightforward and kind of fun. It beats the pants off SNMPv1 for security. So, get going. Get your network moved over to SNMPv3. You will sleep much better.

**One Caveat**

As part of your deployment process, be sure to test the security of SNMPv3. Do not rely on the implementations from your suppliers to be correct. We encourage and support thorough SNMP testing by suppliers, but often they miss the mark. You will want to test for false positives - SNMPv3 agents or managers that accept invalid authentication and/or privacy keys. The SNMP tester should be sure to use a valid key with extraneous characters appended or prepended.
Diagram D illustrates the authentication and privacy processing flow. Proper credentials are required for decryption. If the correct privacy key is not present, the packet cannot be decrypted properly. The key is formed by using a hashing function and two values: the value of the snmpEngineID of the SNMPv3 entity and the value of the localEngineID of the SNMPv3 entity. The combination is used to produce a message digest, which is then compared to the value contained in the message. If they match, access is granted.

One Caveat

Authentication is optional in SNMPv3. There may be situations where the users are trusted and/or the data is not sensitive enough to warrant encryption. This is particularly true in the case of development and debugging. Alternatively, the network administrator may choose to use no authentication to simplify network management. This is called “noauth/nopriv”.

The key to understanding the proper use of authentication and privacy is to consider all of the possible goals. The two primary goals of authentication are to verify the identity of the user and to prevent unauthorized access to the agent. The second part: privacy/encryption

Authentication and privacy are two sides of the same coin. Authentication is necessary for privacy to work. If the sender of a packet cannot be verified, then the packet cannot be decrypted. The most important thing is that the password itself is never sent “in the clear” over the wire as in the case of v1 and v2c. The SNMP tester should be sure to use a valid key with extraneous characters appended or prepended.

Context Name

The second part specifies where the packet payload is coming from and going to. (There will be more discussion on this later.) Each SNMPv3 agent has an engine ID that uniquely identifies the agent in the device. The engine ID may be set at the factory or created by the network administrator. Each module configuration has a context engine ID that is associated with the module or device. The contextEngineID of the entity is used as the contextEngineID in the packet. This allows for multiple instances of the same SNMPv3 entity to be running on the same device.

Reasons to Use or Not Use Privacy

Privacy adds a layer of security by encrypting the packet payload. However, this comes at a cost. There is always a cost in terms of using more network resources. It may be that adding privacy is not worthwhile. The question is whether the added security is worth the added cost. Privacy should be used when the data is sensitive and confidentiality is important. It is possible for logical switch A to fail while the logical switch B continues to run. In a situation like this, the MIB information provided by switch B may not be valid. The network administrator also has to decide what privileges each user has - either individually or as a group. The network administrator configures those users for a full view of the MIB objects and full privileges. Alternatively, the network administrator could define a group called “wheel”, configure the group in this manner, and then simply add Rajiv to the group.

The second part: privacy/encryption

Privacy adds a layer of security by encrypting the packet payload. However, this comes at a cost. There is always a cost in terms of using more network resources. It may be that adding privacy is not worthwhile. The question is whether the added security is worth the added cost. Privacy should be used when the data is sensitive and confidentiality is important. It is possible for logical switch A to fail while the logical switch B continues to run. In a situation like this, the MIB information provided by switch B may not be valid. The network administrator also has to decide what privileges each user has - either individually or as a group. The network administrator configures those users for a full view of the MIB objects and full privileges. Alternatively, the network administrator could define a group called “wheel”, configure the group in this manner, and then simply add Rajiv to the group.

Now, let's take a look at what a network administrator has to do to set up SNMPv3.

Authentication and No Encryption

The user at the management application is authenticated by the SNMPv3 entity. The management application provides the authentication password, and the agent provides the authentication key. The agent generates a message digest using the password and the context engine ID. If the message digest is correct, the user is authenticated. The agent then checks to see if the user is in the group called “admin”. If the user is in the group called “admin”, then the user is granted access. Alternatively, the network administrator could define a group called “wheel”, configure the group in this manner, and then simply add Rajiv to the group.

Thanks

Our thanks to Dave Perkins, Author of Understanding MIBs, and Xiang Li of InterWorking Labs, for their helpful comments on this paper.